e-baton: Recognising Conducting Gestures with Machine Learning

Marko Jeremic
mjeremic@berklee.edu
Berklee College of Music
Electronic Production and Design
Boston, Massachusetts, USA

Akito van Troyer
avantroyer@berklee.edu
Berklee College of Music
Electronic Production and Design
Boston, Massachusetts, USA

Abstract

We present e-baton, a wireless conducting controller that uses an inertial measurement unit (IMU) and a user-retrainable kNN classifier to simultaneously control human performers and electronic musical parameters. During iterative development, we found that accelerometer data alone was insufficient for recognising slower, fluid conducting gestures, and that extending the feature set to include jerk and rotational data was necessary to support the gesture vocabulary used in performance. We evaluate the system through a five-participant tempo-tracking study and a debut performance with an improvising keyboardist.

Keywords

Machine Learning, Conducting, Wekinator, Physical Computing

Figure 1: Exterior and internal images of the e-baton prototype.

1 Introduction

Gestural control of music has been a growing topic in electronic performance. Artists such as Imogen Heap [10, 11] and Susumu Hirasawa [7] have been experimenting with introducing new gestural control into music, providing a more obvious and engaging sound-movement mapping for viewers.

In the domain of electroacoustic music, there has been limited exploration of conducting-style interfaces that allow conductors to interact with electronic elements of a performance. The creation of the e-baton originated in exploring how these conducting gestures can be detected and used to control electronic music. The e-baton uses these gestures to control various parameters within a DAW, such as triggering MIDI clips and controlling parameters of effects.

This paper makes two contributions. First, we report a design observation from iterative prototyping: for IMU-based recognition of conducting gestures, accelerometer data alone proved insufficient, and the feature set required extension to include jerk and rotational data. Second, we demonstrate a wireless conducting interface that preserves the conducting gestural paradigm (Table 1) and integrates with DAW-level control via a per-user retrainable classifier, evaluated through a five-participant tempo-tracking study (section 4) and a debut performance.

2 Previous Work

Composers have been experimenting with the role of conductors since the 1900's. Stratégie (1962) is a composition written by Iannis Xenakis that puts two conductors in a game against each other, using gestures to control their own respective orchestras' playing, gaining and losing points based on the other conductor's response [15]. Other examples include John Zorn's Cobra (1984) and Pierre Boulez's Éclat (1964), where both pieces give the conductor the active role of determining what the performers are playing based on a set of rules determined before the piece [4, 8].

Max Mathews researched the role of the conductor through his creation of the radio baton in the 1980s. The radio baton was an alternative controller that used two batons to map XYZ motion and convert it to MIDI data [14]. Famously, the radio baton was used to conduct the beginning of Beethoven's 5th symphony in a video by Max Mathews. While important for other instruments to follow, the radio baton falls short in its similarity to conducting, breaking away from the typical conducting gestural paradigm. The reliance on a flat surface limits the possible motion, making its use in ensemble environments less viable.

The Digital Baton by Paradiso and Marin was designed to closely resemble a traditional conducting interface, using a 3-axis accelerometer, pressure sensors, and an IR camera to enable 11 degrees of freedom, all of which were mappable parameters for performance. This instrument was frequently used by the MIT Media Lab for a variety of performances, even having pieces written for it [12]. While the e-baton has fewer degrees of freedom, its use of more elaborate gesture detection enables greater control in six degrees of freedom. These six degrees are used to send 10 extracted features to the machine learning algorithm.

The creation of Wekinator by Rebecca Fiebrink has streamlined the process of creating machine-learning-based controllers, with OSC-based integration. Laetitia Sonami used Wekinator for her instrument Spring Spyre, inputting features extracted from hacked reverb pickups using biquad filters in Max, running through a neural network, and outputting audio using PAF synthesis [6].

Most prior research on computer-based conducting tracking relies on computer vision, focusing on practical applications

NIME '26, June 23–26, 2026, London, UK

Table 1: Comparison of e-baton to other electronic batons

Instrument	Communication Protocol	Machine Learning Based?	Wireless?	Preserves C.G.P?
Digital Baton	Serial, Infrared, MIDI	No	No	Yes
Radio Baton	MIDI	No	No	No
e-baton	OSC,MIDI,BLE	Yes	Yes	Yes

such as analysing and correcting a user's conducting or assisting visually impaired choir singers [9].

While computer vision approaches offer marker-free tracking, they require camera setup and controlled lighting [5]. e-baton's IMU-based approach prioritises portability and stage robustness at the cost of reduced gesture vocabulary

3 System Design

Inside a baton case, an XIAO nRF52840 Sense with an inbuilt 6-axis IMU sends data to a subscribed computer using BLE [3]. Although less reliable than other communication protocols, BLE enabled quick prototyping, thereby accelerating the development of other aspects of the controller. This case is a modified version of an existing 3D model found online [2].

The computer runs a Python script that handles the connection and sends data to a central Max patch via OSC. This central Max patch handles the rest of the data transmission, communicating with Wekinator, Ableton, and other Max patches being used for performance via OSC and MIDI.

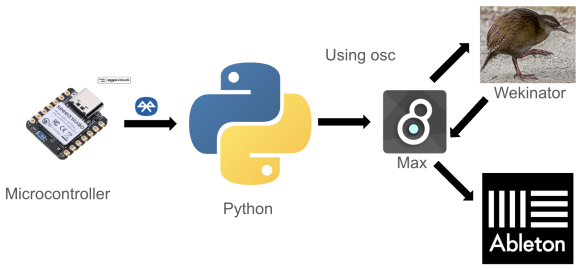

Figure 2: Signal flow of the e-baton system. IMU data streams from the XIAO nRF52840 Sense over BLE to a Python bridge, which forwards OSC to a central Max patch. The Max patch communicates with Wekinator for gesture classification and with Ableton Live for parameter control.

The first gesture trained was to detect beats. This allowed the transmission of timing information to electronic parameters, enabling various performative uses, such as synchronizing Max patches with performers and creating more flexible click tracks. Machine learning was chosen for beat detection because it allows the classifier to be retrained per user, rather than relying on a fixed threshold tuned to a specific gesture style. Rather than using threshold-based peak detection, which requires very specific gestures that may not be accessible to all users, the classifier can be retrained for each user to accommodate variations in conducting style. Wekinator uses a k-Nearest Neighbor algorithm, with 800 examples per gesture class and 500 examples used for beat detection. To train the model accurately, a Max patch was used to send timed OSC messages to Wekinator. These messages were synchronised with an audible metronome that the user would conduct. Data were being sent from the microcontroller to Wekinator at a sampling rate of 100 Hz. OSC messages are used to start/stop Wekinator recording, making training the model on beats a process of conducting to a tempo. To prevent false positives in beat detection, a debouncing system enforces a minimum interval between successive beat events.

In addition to beat detection, Wekinator runs a second classifier that detects changes between parameter banks. The e-baton Max patch provides three parameter banks that can be swapped between, each of which can be mapped to a unique parameter. Each parameter also has an adjustable threshold that can be used to trigger events within other Max patches or control binary parameters. These gestures can be anything the user requires, provided they are distinct enough to be consistently recognised; the ten-feature input vector sent to Wekinator provides sufficient flexibility for this. The input features are: three-axis acceleration, three-axis angular velocity, three-axis jerk, and acceleration magnitude. From these inputs, Wekinator runs two classifiers in parallel: a binary classifier for beat detection and a three-class classifier for parameter-bank selection. Once a parameter bank is selected, the user can move the baton along the XY axis to control two distinct sliding parameters. This is done using a moving average with a window size of 50ms.

For performance, the e-baton patch can be loaded as a Max for Live device, allowing it to control any aspect of the Live API for Max [1]. It can also output OSC messages to other languages, providing flexibility in performance. In its debut performance, the e-baton patch communicated with another Max patch outside Live, controlling triggers for various generative MIDI patches and sending CC messages to Live for MIDI mapping.

A switch on the baton is responsible for three controls: changing parameter banks, recording tempo, and recording parameter changes. This is done in the central Max patch, where the different command inputs can be detected.

The three input combinations for this are a single press, two presses, and a long press. The single press is also used to cancel toggles for recording BPM and parameter changes. After a single button press, there is a 1.5-second window for the user to press the button again for it to be detected as a double press. If the button is still held down after the 1.5-second window, it is counted as a long press.

4 Evaluation/Discussion

We evaluated tempo-tracking accuracy with a five-participant study. Each participant trained the model by conducting to a 120 BPM metronome for 90 seconds, then performed two bars at 60, 90, 120, and 150 BPM while their detected BPM was compared against the metronome reference. Across all participants and tempi, the system produced a mean offset of -0.3 BPM (SD = 2.4 BPM). After brief instruction on improving gesture clarity, all participants conducted within ±1 BPM of the target tempo.

During iterative development, accelerometer data alone produced unreliable classification of slower, fluid gestures. We extended the feature set to include jerk (rate of change of acceleration) and rotational data, after which the gestures used in

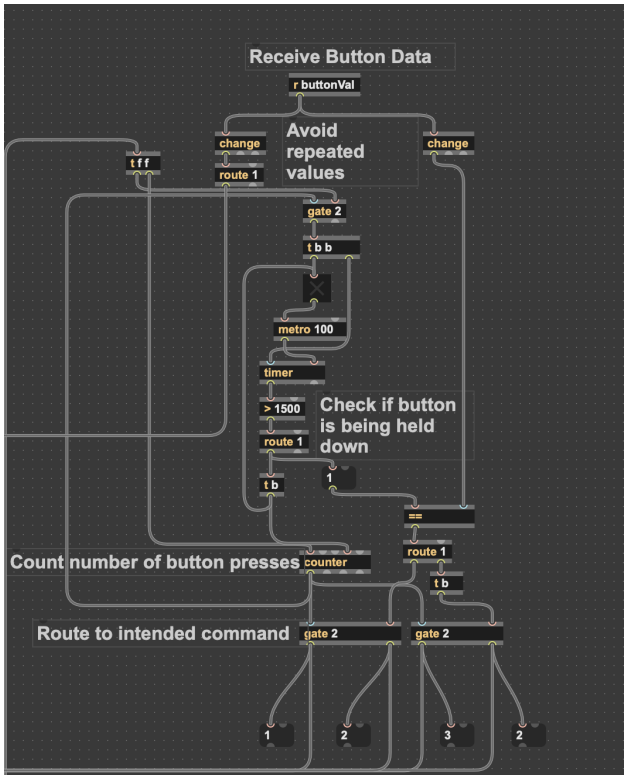

Figure 3: Switch command detection.

the debut performance could be classified consistently enough to support live performance. We did not conduct a controlled ablation comparing feature sets, so the relative contribution of each feature group is a question for future work.

Testing also demonstrated the importance of a UI that displays gesture data so that the conductor is aware of any accidentally misidentified gestures. The UI also displayed which commands were active and the parameter bank levels. Another potential point of failure was the BLE connection dropping, which was addressed by implementing a reconnection mechanism in the Python script used to receive the BLE data.

Figure 4: Max for Live Interface for the e-baton

The biggest limitation came from parameter control. Using a moving average to calculate displacement required precise, rapid movements to avoid unwanted results. The rotations, while beneficial, also caused issues when unwanted; deriving directional components of motion from the rotational data was outside the scope of this prototype.

The debut performance of e-baton featured its use to accompany an improvising keyboard player. The performance was fully improvisational in an ambient style, lasting three minutes, with the only preparation being to familiarise the keyboard player with the conductor's gestures and their sonic mappings. The conductor used triggers to control the activation of generative MIDI patches sent to Live, comprising percussive and melodic elements. All generative patches were synchronised to the conducted BPM. The MIDI from the keyboard player was routed through a piano VST, with various time-based effects controlled by the e-baton, including reverb and delay sends. A substantial practice was required to correlate movements with sounds for the keyboard player, informing them of what to play for specific movements.

End-to-end latency from gesture to Ableton was limited by the switch command system, which required a fixed delay before issuing a command to detect multiple switch presses. Beat detection was perceived as responsive by the performer during the debut performance, though end-to-end latency was not formally measured.

Performing with e-baton led to the development of two performance techniques. With slight rotations, the conductor could adjust parameters without moving in any particular direction or signaling anything to the performer. Being able to toggle recording also helped with conducting the piano player, as it allowed individual control of only their performance rather than also controlling parameters.

5 Conclusion and Future Work

The e-baton explores how conducting-style gestures can be recognised from a six-DoF IMU using a per-user retrainable classifier. Iterative prototyping suggested that feature sets extending beyond raw acceleration, to include jerk and rotational data, were necessary to support the gesture vocabulary used in performance. Future work will include a controlled ablation to quantify the contribution of each feature group, and exploration of alternative communication protocols such as ESP-NOW or OSC over WiFi to address the BLE reliability limitations observed here.

Following review and further improvements to the initial design, the e-baton will be made open-source, enabling users to build their own e-baton and further experiment with the controller's gestural capabilities.

6 Ethical Statement

All testers of the e-baton were informed of their contribution to improving this instrument. Performers provided consent to being recorded for further analysis.

Acknowledgments

To David Cardona and Ryan Page for their unwavering support and resource referrals on this project, and to Kristo Kondakçi for inspiring this project through his conducting. Also, thank you to Berklee and the EPD department for their support.

The e-baton was developed as a final project for an undergraduate course on prototyping electronic digital instruments at Berklee College of Music [13], and this paper reports on the design and initial evaluation of the prototype.

References

[1] [n. d.]. LOM - The Live Object Model - Max 8 Documentation. https://docs.cycling74.com/max8/vignettes/live_object_model

[2] 2019. ConductingBaton. https://cults3d.com/en/3d-model/various/conductingbaton

[3] 2024. Getting Started with Seeed Studio XIAO nRF52840 Series | Seeed Studio Wiki. https://wiki.seeedstudio.com/XIAO_BLE/

[4] John Brackett. 2010. Some Notes on John Zorn's *Cobra*. *American Music* 28, 1 (April 2010), 44–75. https://doi.org/10.5406/americanmusic.28.1.0044

[5] Fahn Chin-Shyurng, Shih-En Lee, and Meng-Luen Wu. 2019. Real-Time Musical Conducting Gesture Recognition Based on a Dynamic Time Warping Classifier Using a Single-Depth Camera. *Applied Sciences* 9, 3 (Feb. 2019), 528. https://doi.org/10.3390/app9030528

[6] Rebecca Fiebrink, Dan Trueman, and Perry R. Cook. 2009. A Meta-Instrument for Interactive, On-the-Fly Machine Learning. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Pittsburgh, PA,

United States, 280–285. https://doi.org/10.5281/zenodo.1177513

[7] hirasawalyrics. [n. d.]. Susumu Hirasawa Interview (2009). https://hirasawalyrics.tumblr.com/post/81459318825/susumu-hirasawa-interview-2009?utm_source=chatgpt.com

[8] Dominique Jameux. 1991. *Pierre Boulez*. Faber and Faber, London.

[9] Noriyuki Kawarazaki, Yuhei Kaneishi, Nobuyuki Saito, Takashi Asakawa, Kanagawa Institute of Technology, 1030 Shimo-Ogino, Atsugi-shi, Kanagawa 243-0292, Japan, Asap System Co., Ltd., Setoru Higashioume 109, 2-18-5 Higashioume, Oume, Tokyo 198-0042, Japan, and Kinki University Technical College, 7-1 Kasugaoka, Nabari, Mie 518-0459, Japan. 2014. A Supporting System of Choral Singing for Visually Impaired Persons Using Depth Image Sensor. *Journal of Robotics and Mechatronics* 26, 6 (Dec. 2014), 735–742. https://doi.org/10.20965/jrm.2014.p0735

[10] Thomas Mitchell and Imogen Heap. 2011. SoundGrasp : A Gestural Interface for the Performance of Live Music. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Oslo, Norway, 465–468. https://doi.org/10.5281/zenodo.1178111

[11] Thomas Mitchell, Sebastian Madgwick, and Imogen Heap. 2012. Musical Interaction with Hand Posture and Orientation: A Toolbox of Gestural Control Mechanisms. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. University of Michigan, Ann Arbor, Michigan. https://doi.org/10.5281/zenodo.1180543

[12] Teresa Marrin Nakra and Joseph A. Paradiso. 1997. The Digital Baton: a Versatile Performance Instrument. In *International Conference on Mathematics and Computing*. https://api.semanticscholar.org/CorpusID:31481523

[13] Akito van Troyer. 2026. From Controller User to Instrument Designer: Teaching NIME in a Contemporary Music Context. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. London, UK.

[14] Bill Verplank, Craig Sapp, and Max Mathews. 2001. A Course on Controllers. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Seattle, WA, 7–10. https://doi.org/10.5281/zenodo.1176380

[15] Iannis Xenakis. 1972. *Formalized music: thought and mathematics in composition* (2. print ed.). Indiana Univ. Pr, Bloomington, Ind.